Table of Contents

1. EXECUTIVE SUMMARY

Encyclopedia Britannica defines the word engineering as ‘‘the application of science to the optimum conversion of the resources of nature to the uses of humankind’’. In parallel to this definition, throughout history; to do research; to design, invent and manufacture products in order to satisfy our everlasting needs and demands within the restrictions of our limited resources have been among the main purposes of engineering. Consequently, our expanded horizons have led us to feel the need to have robots that will do the hard work instead of us, and maybe, for us.

In the last decade, as the robot systems became more and more widespread, autonomous devices have experienced a rise of attention. The applications of autonomous devices are numerous. Therefore, studying these concepts and designing and manufacturing a product that can achieve this, are among the needs of today’s academic and industrial world.

Through this document, JCN, a newly established company aiming at simple, yet elegant solutions to real life problems according to the needs of their customers, presents their main solution steps: Their conceptual design for the problem of constructing a robot that will clear its own area off undesirable objects.

The final product incorporates several subsystems such as motion system, field and position system, target and fence detection system, catching and throwing system and power system. The philosophy of the design is based on simplicity and robustness, as a commercial product would be restricted mostly by these specifications. The proposed system comes into prominence with its detection system and its effective catching and throwing system as the rest of the report will show.

JCN has been working on this project extensively and continuously for the last 7 months. To design and manufacture such systems require a substantial amount of knowledge about signal processing, control theory and robotics. Therefore, JCN, as a company proud to work on such a project and each and every team member is happy to contribute to it. Not only is this project a road to synthesize the theoretical knowledge they have acquired so for but also, an opportunity to learn a lot more.

2. BRIEF INTRODUCTION

Recently founded by five ambitious engineers, JCN is a company focusing on robotics, control systems and digital signal processing. Each of these five group members has different major fields of study and they encompass a wide range of professional interests. Believing that the interdisciplinary study is an integral part of their problem solving process, JCN is glad to have this diversity in the company.

For their first project, JCN is now working on to design and build ‘‘A robot which clears its own area off undesirable objects.’’ In this project, JCN provides their customers with a Servant robot, adjustable field and adjustable objects, along with the deliverables that come with the end product. The main operation of the product is to detect the undesirable object with the help of the sensors, catch it and throw it to the other field without colliding the fence or the other robot. The design basically consists of the following subsystems:

- Microcontroller subsystem consisting of two Arduino Nano to integrate all individual subsystems under a complete system,
- A power subsystem to distribute the power to the product efficiently,
- A motion subsystem consisting of the main body of the vehicle, motors and motor driving circuitry,
- A field detection subsystem consisting of four CNY sensors at the corners of the robot to detect the borders of the field,
- An object detection subsystem consisting of two ultrasonic sensors to detect the targets (undesirable objects) and the fence,
- A catch and throw subsystem consisting of a creative mechanic design working with one servomotor and 2 stepper motors.

The main aim of this report is to give the detailed information about JCN’s final product for its latest project. In this report; after presenting the company and briefly explaining the main system, each sub-block has been presented. Technical details about the product, specifications and test results have been given. Moreover; deliverables and overall cost of the product have been discussed.

3. COMPANY ORGANIZATION

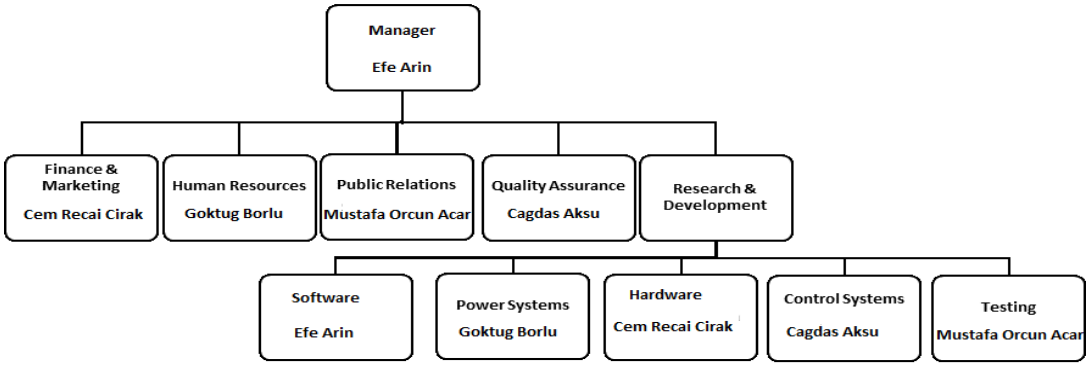


Figure 1: Company Organization

Hata! Başvuru kaynağı bulunamadı. shows the Company Organization. In this Figure, the divisions are done using the conceptual relationship in the company. All five members have responsibilities in the Management and Research & Development parts.

4. DESIGN DESCRIPTION

JCN has designed the robot having 4 sub-blocks, which are intended to function for specific jobs to meet the all requirements of the project. The sub-systems perform individually, and they are controlled by common or different DC motors and microcontroller.

Main Requirements:

- Robots should not exceed the boundaries and fence
- Speed and detection is important because of the limited time and opponents.
- No hard wiring is allowed cross the field boundaries.

Before going to next step, main hardware components of the servant project are;

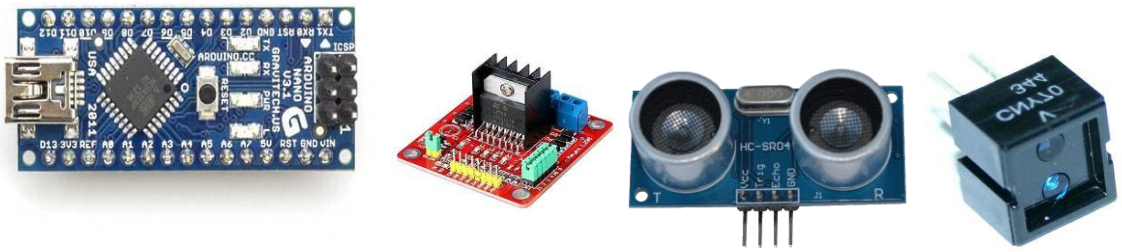


Figure 2 Main components of the Servant

4.1 System Modules

Motion System and Microcontroller: This system is basic response to movement of robot. There is not complex as other three subsystems but it is important as them. Therefore JCN has spared necessary time to develop this system. The purpose of this system is to enable the robot move stably when makes detection and disposing. This system contains two high-torque DC motors with their wheels and one dummy wheel

For microcontroller part, JCN decided to use two separate ardunio nano to perform multiple task functionality in inexpensive way. Why is this way inexpensive? The answer is quite open; for multitasking functionality; high –level microcontrollers such as FPGA should be used. It is already known that these type microcontrollers are so expensive when compared with two ardunio nano.

One of the ardunios is used to perform motor driving, target and field detection (takes the data coming from ultrasonic and contrast sensors).Other one's usage purpose is controlling catch and throw mechanism. The advantage of using this method is that we divide the controls of two main subsystems to two different microcontrollers. By using this way when the servant throws the garbage, it can continue to detection facility.

Detection System: Since the project is mainly based on the detection of the wastes, much more time was spend and also much more research was made for this part of the system modules when is compared other three system modules. At the beginning of the project many solutions were thought such as image process, ultrasonic sensors etc. by shareholders of JCN. At the end much more economic and easy way were determined that using ultrasonic sensors and for this system module. In addition to this for the field detection contrast sensors are used.

Catch and Throw System: This sub-system is designed to send the wastes to opponent side. This system module is second important part of the whole system. Design consists of two alternative solutions according to the distance of servant to fence. If the servant is close enough to fence, then it uses drop functionality otherwise it uses throw mechanism.

Power System: When this module was designed, JCN considered two main things: power consumption and high sensitive to environment. Safety is also important but in these days most of the batteries types can meet this requirement. According to these consideration power system has designed and completed

4.2 System Algorithm

The individual sub systems will be discussed in detail analytically and critically, however to give the initial and basic logic of the device functioning to the customers, ASMD type of system algorithm can be examined as in Figure 3.

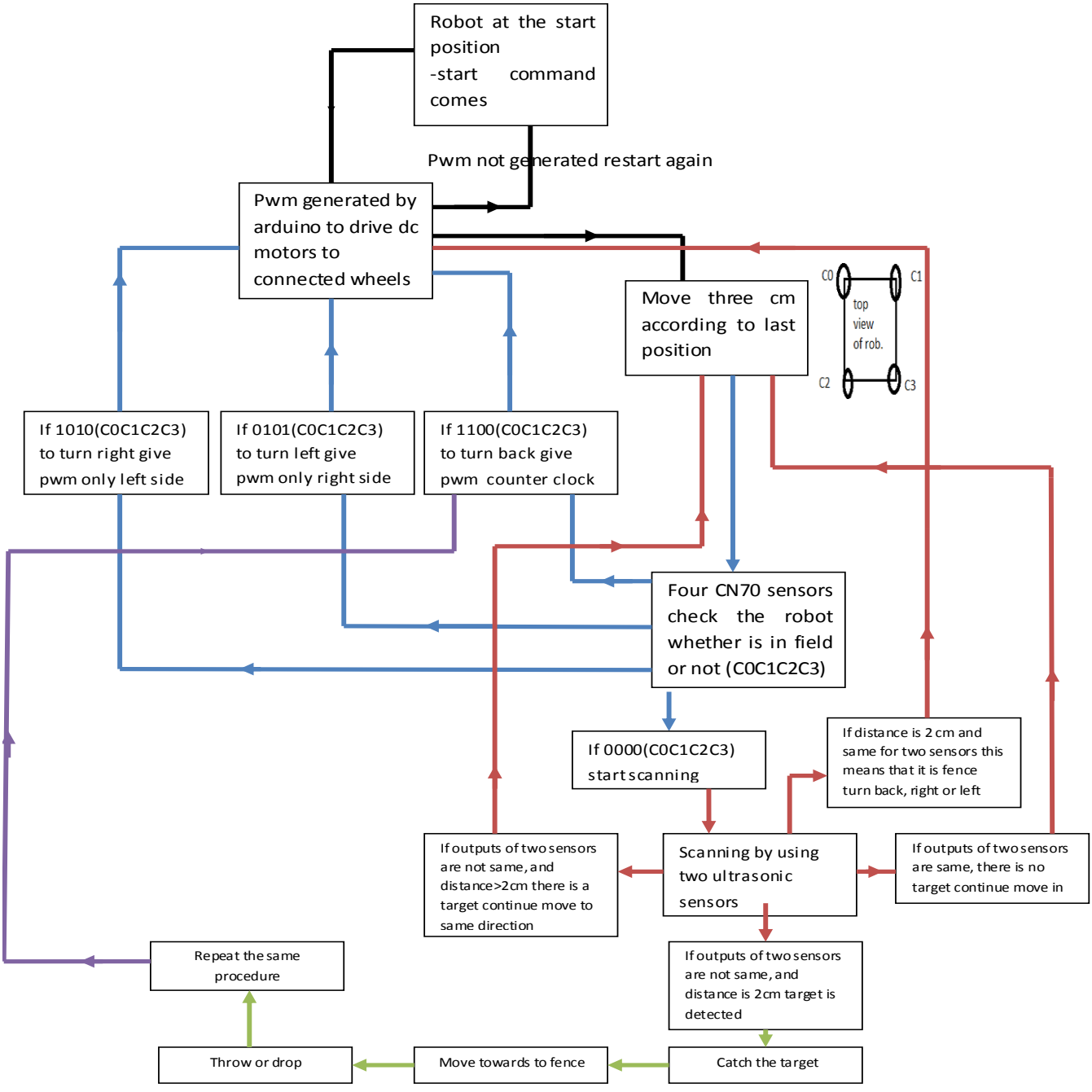
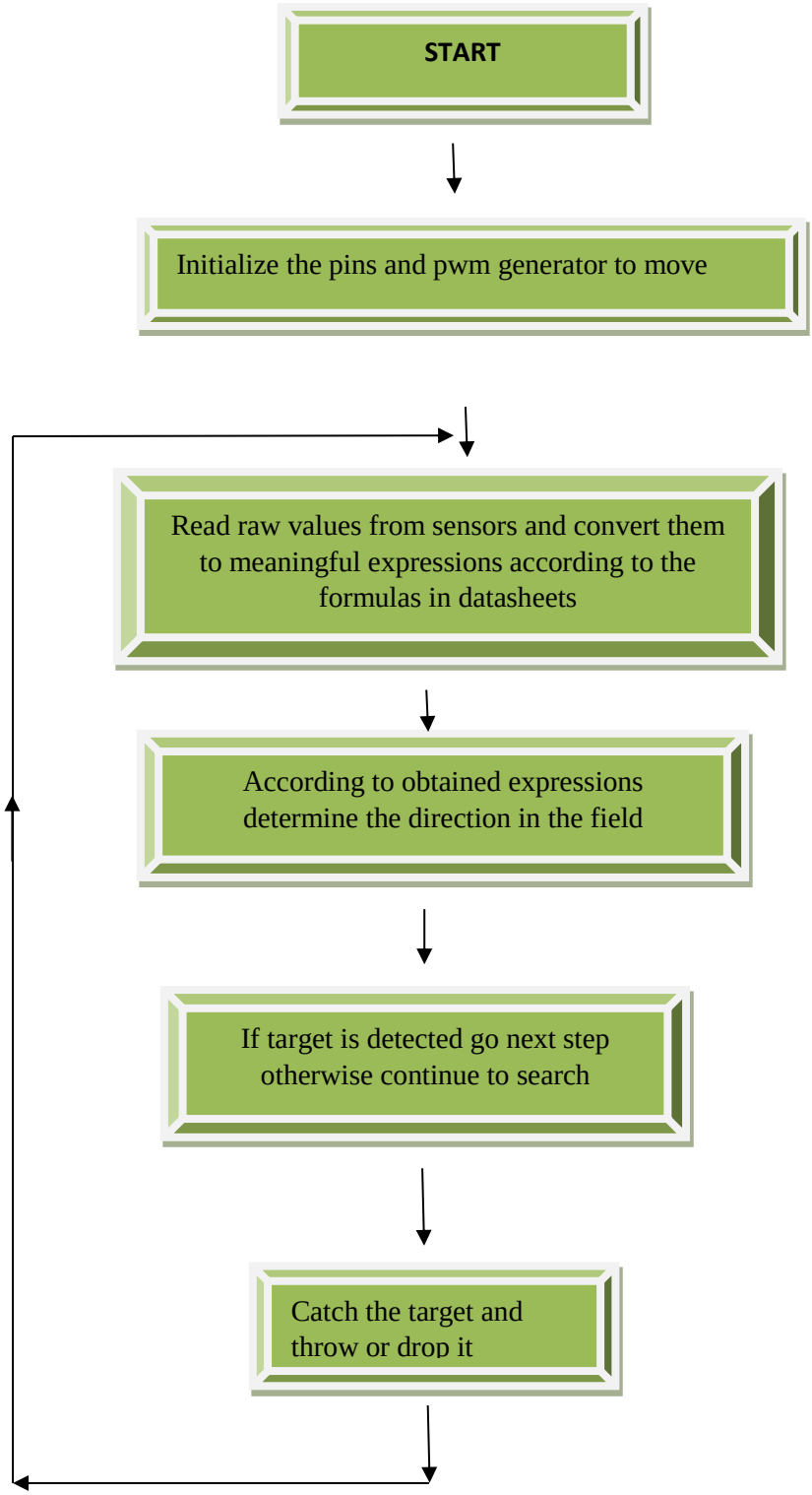


Figure 3 ASMD chart of general design

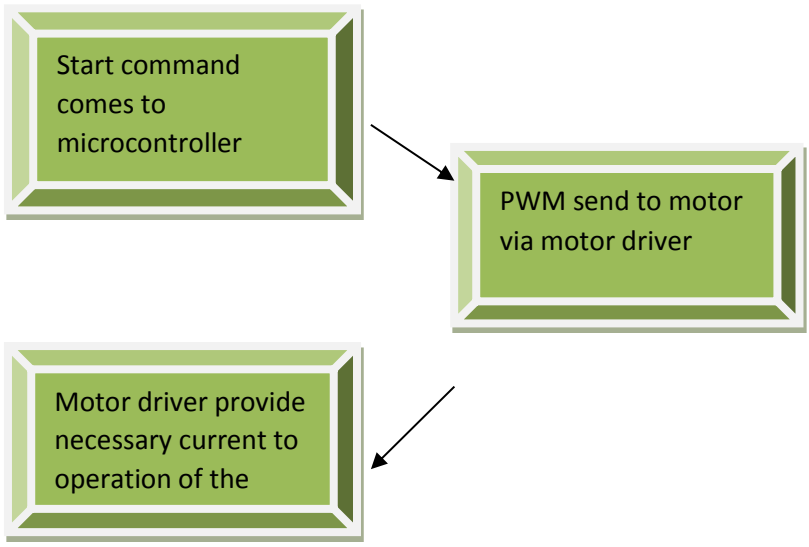
5. ANALYSIS OF THE SUBSYSTEMS

The design of the project is composed of 4 sub-blocks, which are motion subsystem, detection subsystem, catch and throw mechanism, finally power system. Before giving detailed information about these systems, you can see the flow diagram of the servant project as below:



5.1. Motion and Microcontroller Subsystem

The general solution logic can be seen from the chart below.



L298N Motor drive:

Due to that the motor will be requiring more current than the microcontroller pin can provide, this unit is needed for motor driving. This chip contains dual Full Bridge, which accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors.

The L298N is designed to provide bidirectional drive currents of up to 4 A at voltages to 46 V. The L298N provides low saturation voltages and over temperature protection. In addition to this, it provides logical “0” input voltage up to 1.5 V in other words it has high noise immunity.

The L298N IC receives signals from the microprocessor and transmits this signal to the motors. It has two voltage pins, one of which is used to draw current for the working of the L298N and the other is used to apply voltage to the motors. The L298N switches it output signal according to the input received from the microprocessor. For example: If the microprocessor sends a 1(digital high) to the Input Pin of L298N, then the L298N transmits a 1(digital high) to the motor from its Output Pin. An important thing to note is that the L298N simply transmits the signal it receives. It does not change the signal in any case.

Detailed information regarding this IC can be found in given data sheet as an appendix. The pin configuration of the L298N can be seen on Figure 4.

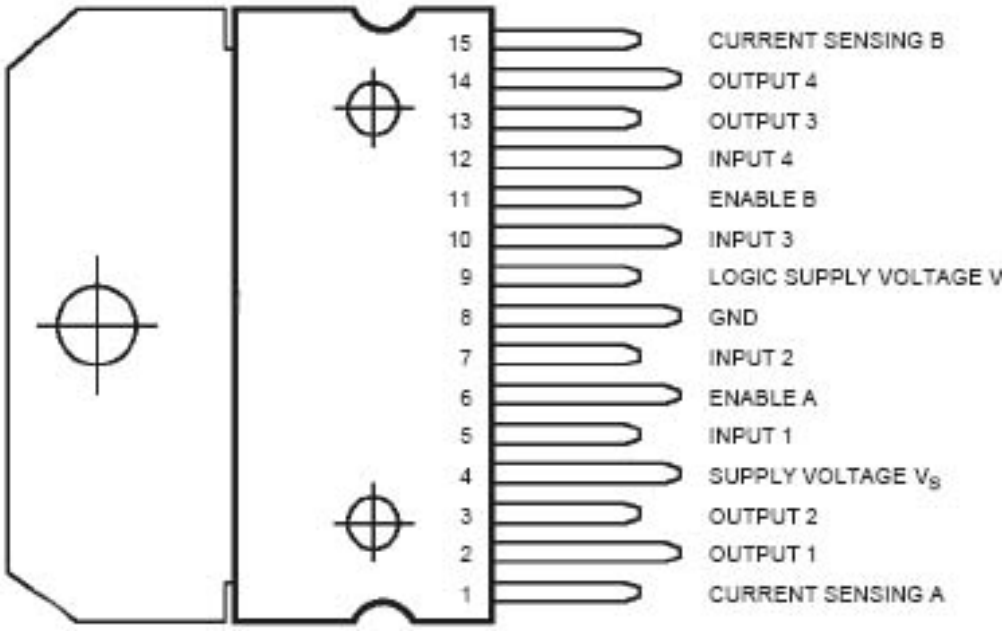


Figure 4 Typical L298N motor drive IC

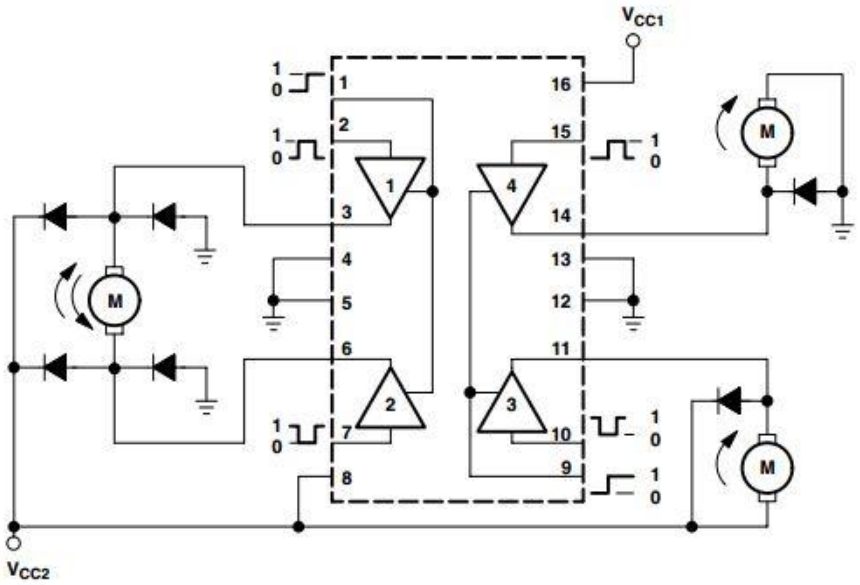


Figure 5 Block diagram of L298N

Since the speed of the wheels of the robot will be controlled (two DC motors) using a microcontroller through the feedback from distance sensing sensors, the overall circuit diagram can be shown in Figure 6.

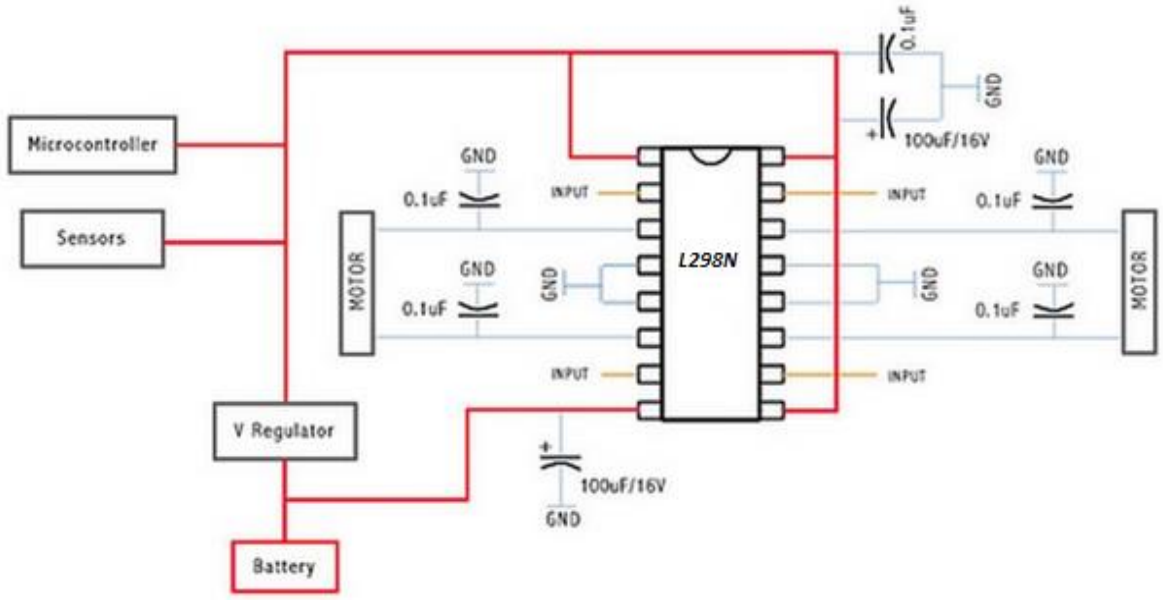


Figure 6 Circuit diagram of motor driver

According to sensors data microcontroller sends a command (pwm) to driver to made scanning process on the given field. Detailed information will be given in detection part but basic principle of the turn right or left based on given pwm to left and right DC motors. In other words when pwm is only given to right motors, the servant turns left side otherwise turns right.

Microcontroller & PWM:

The speed for the following mode of operation will be controlled by a well-known technique of Pulse Width Modulation (PWM). Pulse width modulation is a method for reducing the amount of power delivered to a DC motor. Instead of reducing the voltage operating the motor (which would reduce its power), the motor's power supply is rapidly switched on and off. The percentage of time that the power is on determines the percentage of full operating power that is accomplished. This type of motor speed control is also easier to implement with digital circuitry. It is typically used in mechanical systems that will not need to be operated at full power all of the time.

The first sub system to be implemented in the project was this sub block (motion system), and JCN has completed this part with no error rate. This has been achieved thanks to the successfully planned conceptual design of this sub system, as discussed in detail in the test procedures & results part of this report.

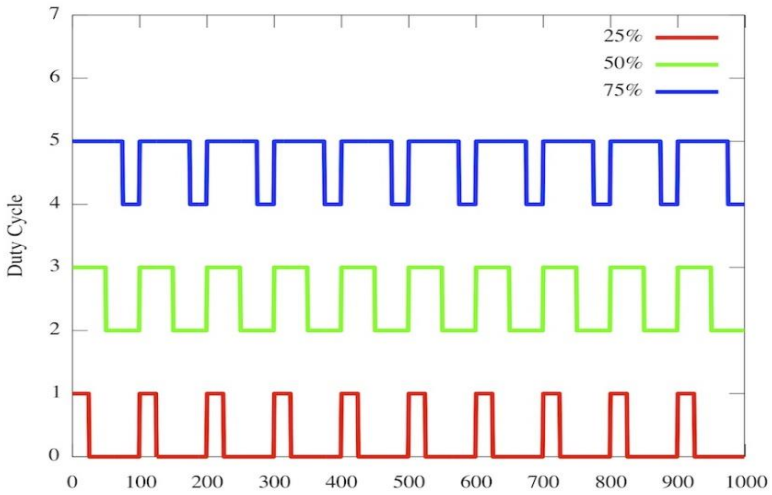


Figure 7 PWM with different duty cycle.

To implement this method, Arduino has been preferred among other types of microcontrollers due to the advantages it offers, some of which can be examined as below;

- Inexpensive

- Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50. Due to these information, it was logical to use Arduino for cost minimization purposes.

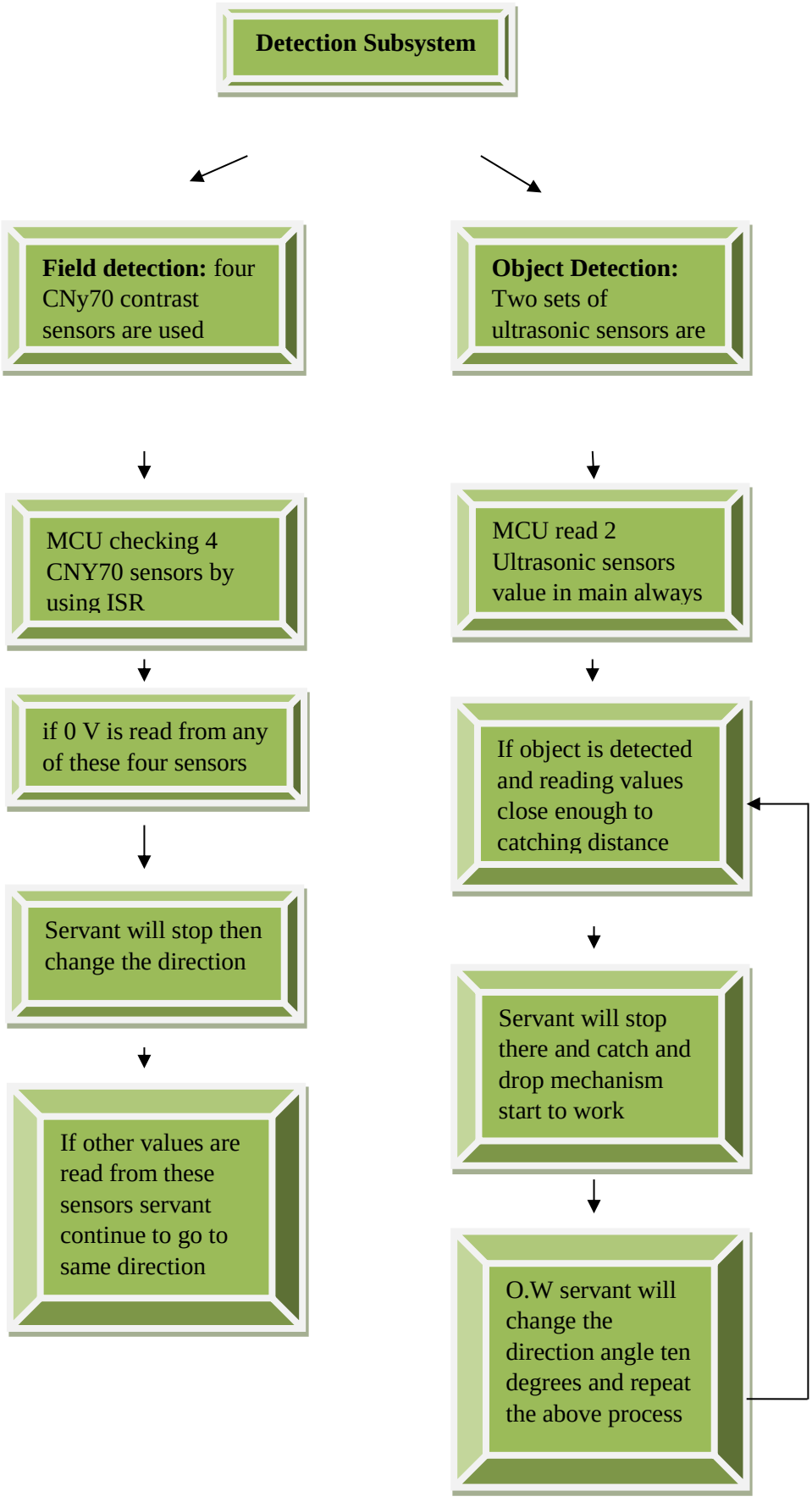
- Cross-platform

- The Arduino software runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.

- Simple programming environment

- The Arduino programming environment is easy-to-use, and flexible, which reduces the amount of time allowed for the coding.

5.2. Detection Subsystem



Detection subsystem was divided in two main blocks. These blocks are field detection and target detection.

Field detection subsystem: is used to identify field by remaining within the field borders and to specify the position of the robot and the direction that it fronts on to. In this part, mainly two CNY70 contrast sensors are used. These sensors are emplaced on the leading edge of robot base 1 cm above the field floor as one on the leftmost and one on the rightmost. In addition, for the sake of to ease the detection process, floor of the field is designed in white color with 2.5 cm width black borders to make a contrast between field and borders. During in motion, the robot checks analog data came from both CNY70 sensors continuously and interprets it as digital data by comparing it with a threshold value which is decided according to experimental measurements. (Using analog data provides capability to make a reasonable comparison with an adjustable threshold value for deciding the contrast level in different environments by neutralizing the undesired distractive environmental effects.) In general case, values read from both sensors are expected under the threshold. Remaining all three cases are considered as primary emergency cases and less than one of these cases related immediate action algorithm is run. These algorithms stern to the robot into the field back and end the emergency case in the shortest time possible. Brief description about CNY70; is a reflective sensor that includes an infrared emitter and phototransistor in a leaded package which blocks visible light. You can see CNY70 and test configuration from Figure 8.

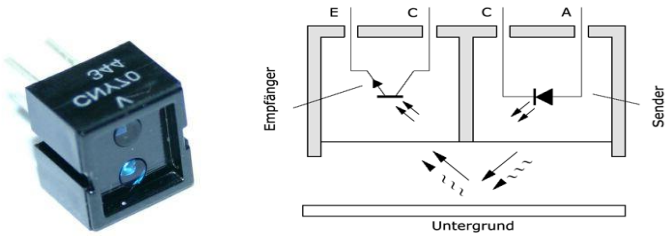


Figure 8 Working principle of contrast sensor

Object Detection Subsystem: Object detection subsystem is used to search and detect the targets and the fence. This subsystem consists of two nested parts as fence detection and target detection. In this subsystem, mainly two Ultrasonic sensors are used. The working principle of ultrasonic sensors is given on Figure 9.

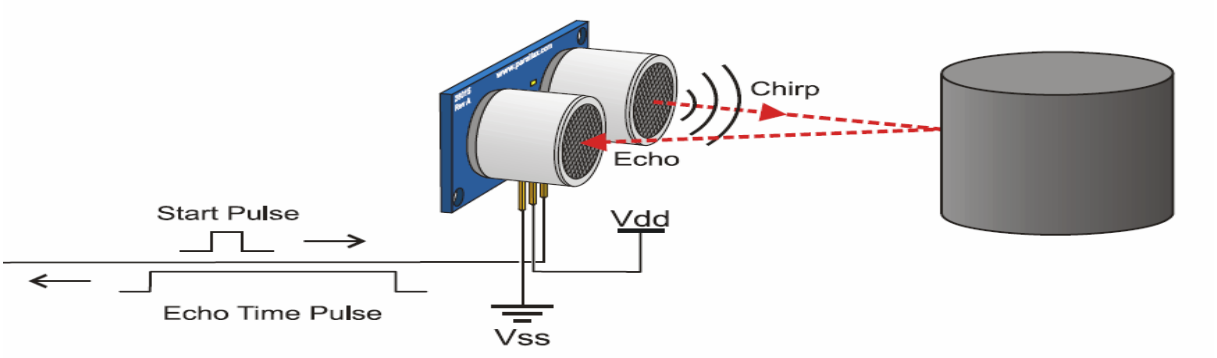


Figure 9 Working principle of ultrasonic sensor

Object detection subsystem also consists of two separate parts:

Fence Detection: Fence detection part aims to avoid possible crashes between the robot and the fence while searching for targets and to detect and to close through the fence with the right angle after a target was captured. The robot decides that the fence is detected if the higher sensor data implies that there is an object.

Target Detection: Target detection part aims to search and to find the targets. In this part, there are three possible cases. First, if the lower sensor implies that there is no object, then the robot decides that there is no target in the scope. In this situation servant change direction angle and scan again object in the range of ultrasonic sensors. This process is continued until the object is found or 360 degrees are completed. If object cannot found in this process servant continue to movement in random direction 2 seconds then repeat the same process. Second, if both sensors imply that there is an object and their data values approximately the same, then the robot decides that there is no target in the scope and the fence is detected. Third, if both sensors imply that there is an object and their data values strictly different (expectedly data value comes from the lower sensor is greater), then the robot decides that a target is detected. When target is detected MCU control the distance whether garbage closes enough to catch mechanism. If it is close catch and drop/throw mechanism will start otherwise servant go to object until the sufficient distance will be caught

5.3. Catch and Throw Subsystem

Before going deep in this subsystem, Figure 10 and Figure 11 are given in order to help you to construct the real mechanism in your brain before reading detailed explanations.

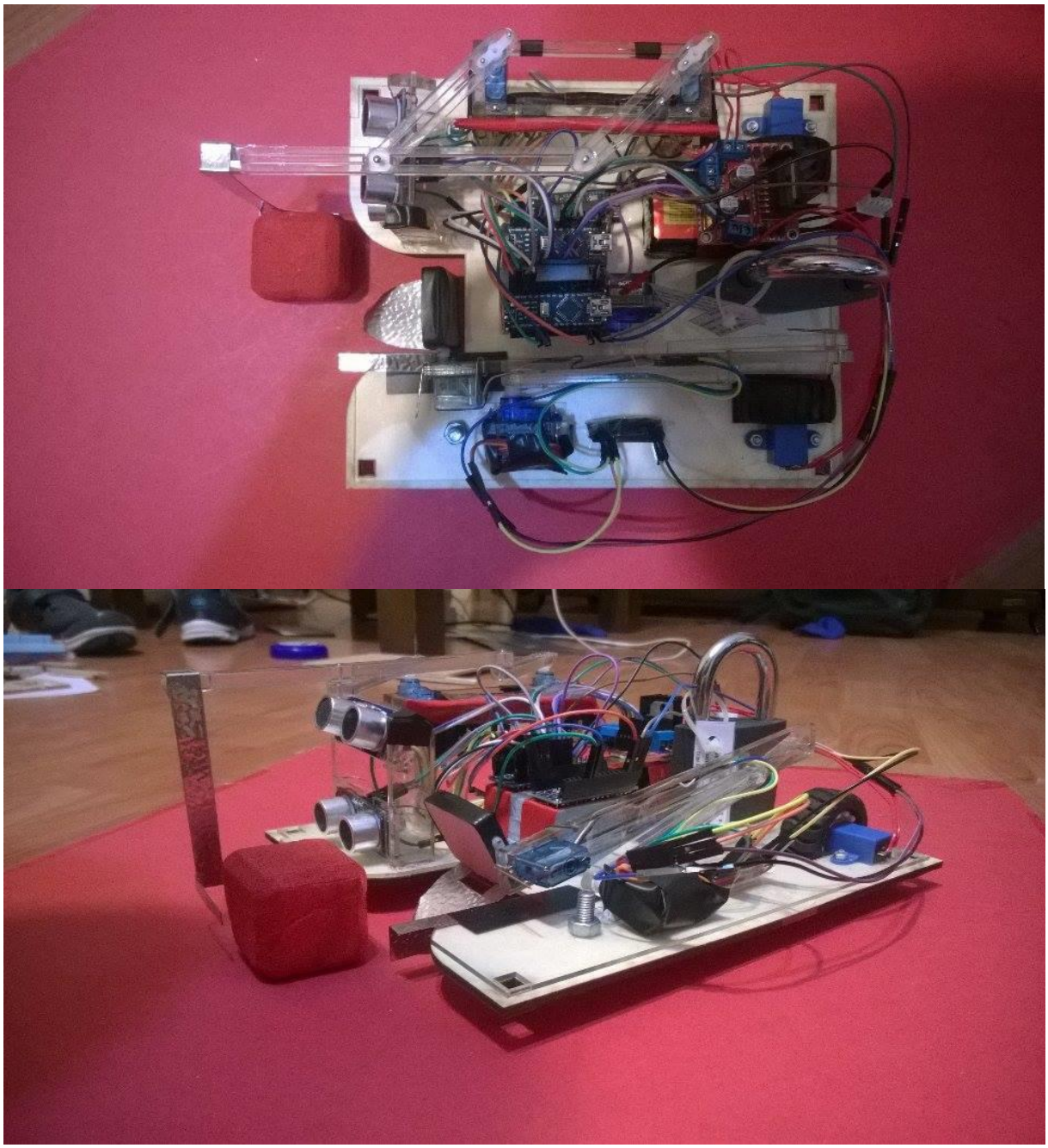


Figure 10

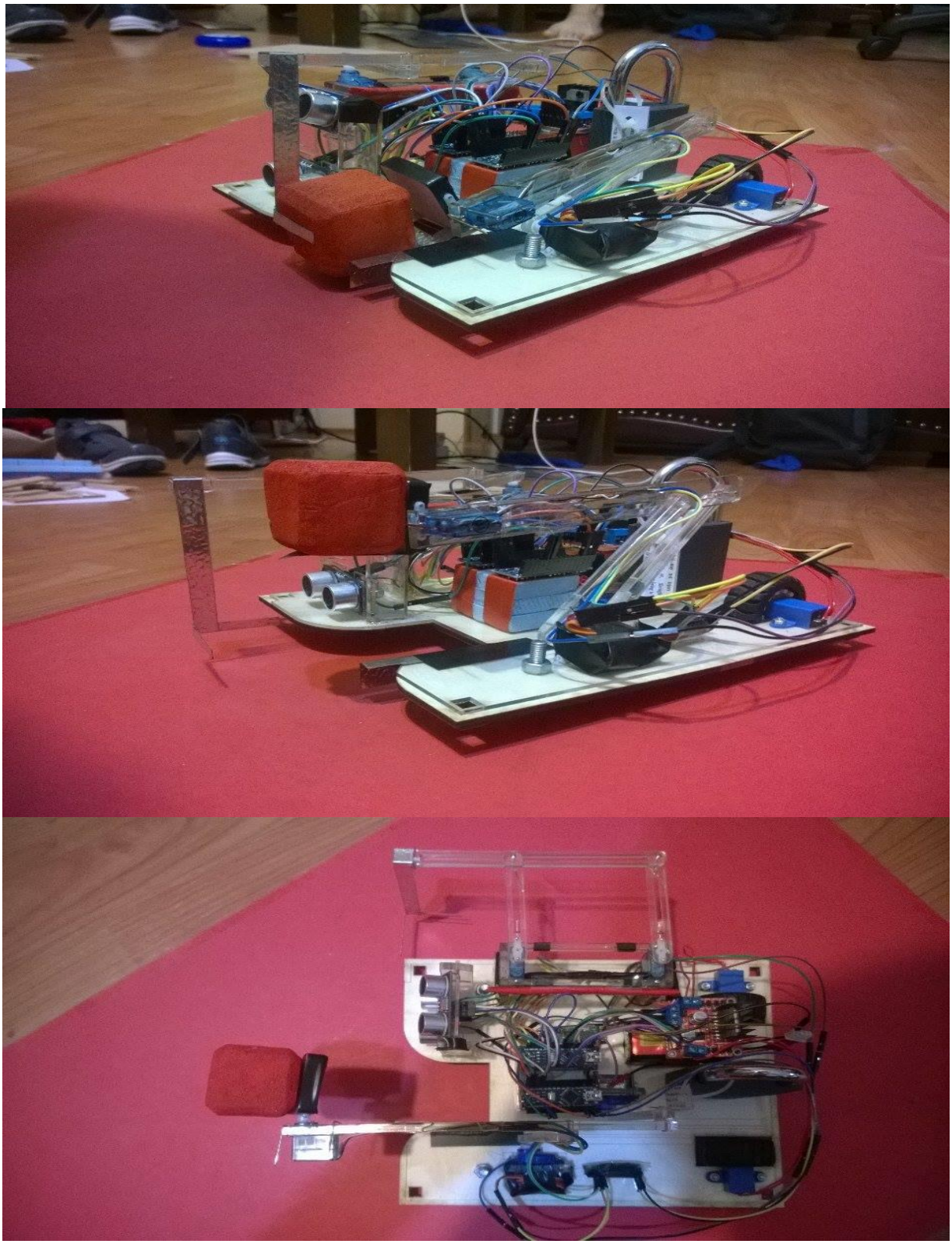


Figure 11

As you can see from Figure 10 and Figure 11, detailed catch and drop/throw mechanism described step by step with given photos. Firstly, servant approaches the target until to exact catch distance is caught. When this process complete, MCU send the start command to motors of catch arm then catch arm opened to take to garbage to lift platform. When catch arm is closed, coming order is send to lifting arm's motors. Garbage is lifted up then drops it over the fence.

Catch: As a catch mechanism The Servant will use a simple arm that sweeps the garbage through the platform which lifts the garbage over the fence for the throw or drop mechanism after detection of garbage The Servants goes through it and servo completes one cycle. Thanks to the pin located to the catch arm and which moves inside the circular rail, garbage will be swept through inside of the robot.

Drop: After catching the garbage 'The Servant' will drop the garbage by moving to just behind the fence. The Servant has a simple, single servo motor controlled drop mechanism. Basically servo just lifts the garbage. Since there will be a spring between platform and lifting arm the garbage will stay aligned through the direction of lift arm up to pin which is fixed to the chassis. Whenever the front point height of the platform reaches to the pin, the platform alignment will start to be destroyed since the front point stops rising while the platform-lifting arm connection keeps rising. It is expected that before the lifting arm-platform connection reaches to its top point the garbage slips and drops down.

5.4. Power Subsystem

The power system consists of three main subunits; the battery, the battery charger and the cables that carry the power. The cables of the product are standard robotics cables and the currents that are carried in the device are insignificant and will pose no overloading problems. The charger is another simple equipment related to this subsystem and works well with the battery. Instructions for the charger can be found in the user manual given in the appendix. The most critical technology in the power system is the battery.

The battery included in the device is a 3-cell, 12 V, 1100 mAh Li-Po battery.

Table 1: Power Consuming Devices in the Robot

Device	Usual Current Demand (mA) at 5 V	Max. Current Demand (mA) at 5 V
Arduino Nano x2	~100 mA (5 V)	Same
Ultrasonic Sensor x2	5 mA (5 V)	Same
Contrast Sensors x4	10 mA (5 V)	Same
Motor Driver	1.5 mA (5 V)	2.2 mA (5 V)
DC Motor x2	140 – 320 mA (5 V)	320 mA (5 V)
Servo Motor x5	20 mA (5 V)	140 mA (5 V)
Total	631.5 – 991.5 mA (5 V)	1592.2 mA (5 V)

6. TECHNICAL DETAILS AND TESTS

Testing the equipment is the critical point for a design project. For specific equipment, tests should confirm the usability and the compatibility for the whole project. JCN has conducted several tests in different time periods of the overall design process. Some of them were performed more than once along the design project. The tests that the JCN has conducted were related to the subsystems and their components.

6.1 Detection Subsystem

For the detection subsystem, ultrasonic and the contrast sensors are used. For the fence and waste detection two imbricated ultrasonic sensors were used. They are placed with an angle of 10 degree between the vertical axis in the Figure 2.

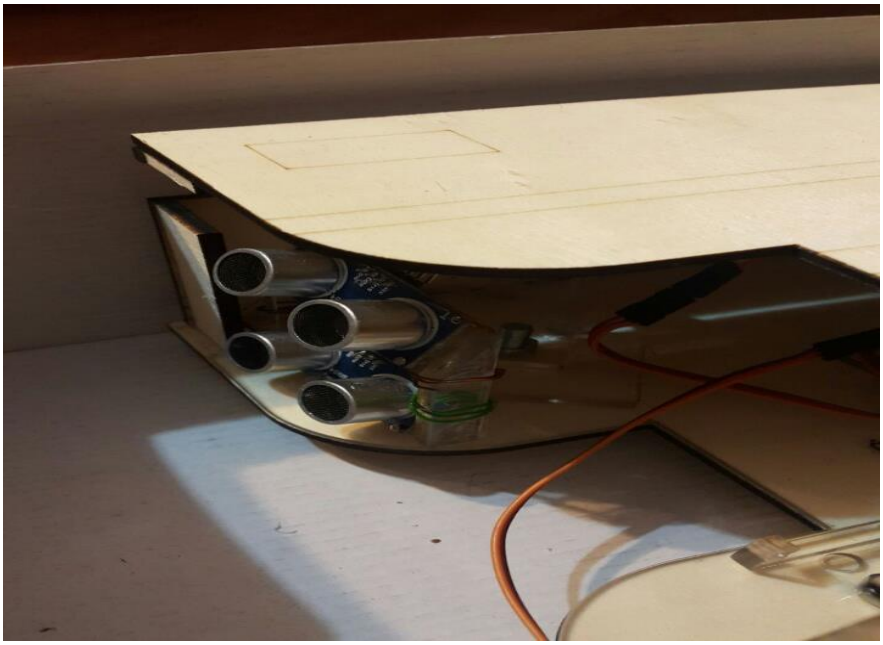


Figure 12 Placement of the ultrasonic sensors

In order to test the ultrasonic sensors, objects are placed at different distances and results are observed. After several trials, it is obtained that those sensors are able to distinguish four-centimeter-particles from 80 cm distance. This was a sufficient result for the gardener project considering the standards for the fence and the waste. Finally they are tested with the field and the wastes. The problems seen during the fence detection are solved by adjusting the algorithm.

On the other hand, CNY 70 contrast sensors, which can be seen on Figure 2, are used for the field and the border detection. Four of them are placed at the corners of the robot. The working principle of a contrast sensor is giving different output signals according to the field under it. Basically, it determines whether the field under it is black or white.

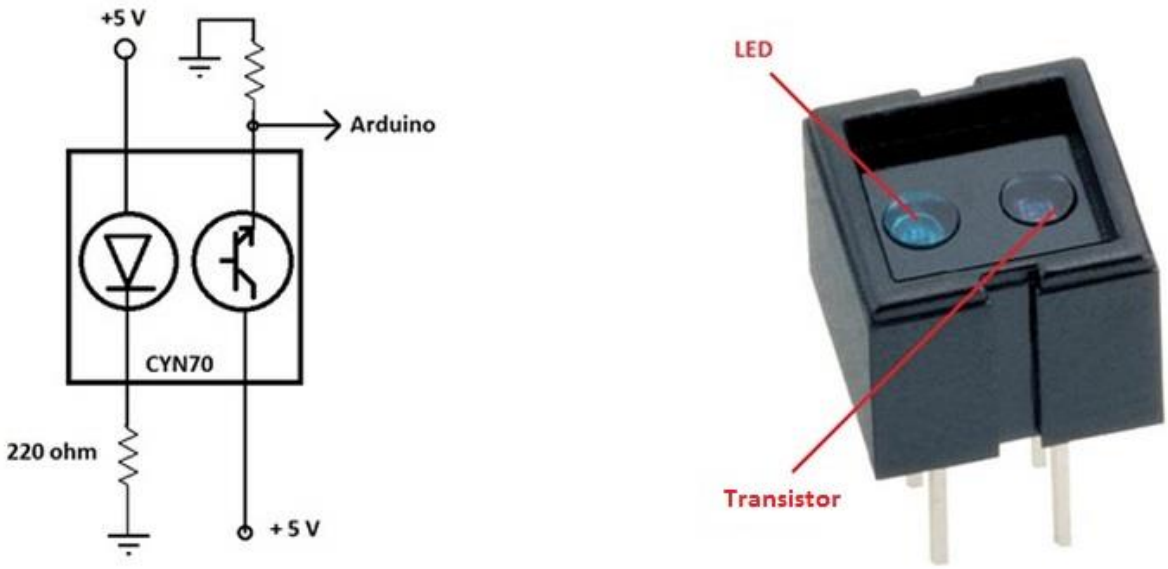


Figure 13 CNY 70

Those CNY 70 sensors are tested with the motion system. Results were satisfying since the robot was able to stay in the field by returning when a black border is detected.

6.2 Catch and Drop Subsystem

There are 5 servos and 1 DC motor in this subsystem. Two of the servos (4.3 gr, 0.8 kg/cm) are for catch system, one of them (4.3 gr, 0.8 kg/cm) is used for drop mechanism, and other two of the servos (9 gr, 1.7 kg/cm) are used for lifting mechanism

Servo motors should be tested in terms of torque and angle. Because of the lack of the necessary equipment such as torque wrench, those tests were conducted by the hand and observations, and the results were promising at first. After the implementation of the robot, it is seen that the subsystem works properly with the real wastes.

6.3 Motion Subsystem

The acceleration and speed control of the vehicle is an important issue. The vehicle should be able to move to search and grab the target and stop without crashing to the waste or the fence. Moreover, the speed of the vehicle should vary while searching and carrying. Actually this can be achieved with the control of DC engines which are used to move the vehicle. This control is satisfied by the PWM method. By the PWM method, the average voltage sent to motor, thus the speed of the motor is adjusted.

DC motors were tested with additional weight in order to implement the actual weight at first and with the real components then. In those tests, it is seen that the robot can travel 2 meters in approximately 10 seconds which corresponds the 90rpm with the wheels which have 4.2 cm diameter. Results showed that DC motors can satisfy the needs of the project in terms of speed.

6.4 Communication Test for Two Arduinos

In the servant project two arduinos were utilized. While one of them is responsible of DC motors used in motion subsystem, the other one is managing the catch and drop mechanism. The main problem of two arduinos working together in a project is the floating problem. JCN developed a protocol in order to prevent that problem to occur.

7. COST ANALYSIS

Budget limitation for The Servant project is $ 150. Approximate cost analysis per The Servant can be seen in the Table 1.

Table 1: Approximate Cost Analysis Per 'The Servant'

Equipment	**Quantity**	**Total Price ($)**
Microcontroller	2	4.00
Battery	1	8.09
DC Motor Driver Module	1	1.75
DC Motor + Wheel	2	5.04
Free Wheel	2	3.00
Stepper Motor	2	2.88
Stepper Motor Driver Module	2	1.94
Servo Motor	1	1.72
Contrast Sensor	2	0.60
IR Distance Sensor	2	8.58
Accelerometer	1	0.99
Blank PCB	1	0.96
Construction	1	10.00
Field + Fence	1	10.00
User Manuel	1	0.10
Steampunk Gear Set	1	2.00
Evil Eye Talisman	1	0.10
Lace	1	1.50
Cable + Soldering Wire		1.50
Others		2
Total		**66.75**

8. DELIVERABLES

The Servant box will include following items:

- The Servant robot itself
- Field
- Fence
- 3 x Target
- Evil eye talisman
- Lace
- User manual

9. APPENDIX

1. USER MANUAL

“The Servant” package includes:

- The robot equipped with 2 DC motors, 2+2 caster wheel system, 4 contrast sensors at the corners, 2 ultrasonic sensors, a Li-Po battery, Arduino Nano.
- 3 targets
- Battery charger

In case of any missing parts please contact to the supplier.

INSTRUCTIONS

- Charge the Li-Po battery with the provided battery charger. Set the current value in the range [0.5A, 1A].
- There is a cable available for battery charging.
- Set the field and fence.
- Set the robot on.
- When the level of battery is low, the robot sounds for warning. Then make the robot off and charge it again.
- DO NOT CHARGE the robot while the robot is on.

MALFUNCTIONS

There are no recorded malfunctions of this product.
In case of any malfunctioning of the parts (sensors,wheels..etc.) please contact to the supplier. DO NOT fix the damaged parts by yourself

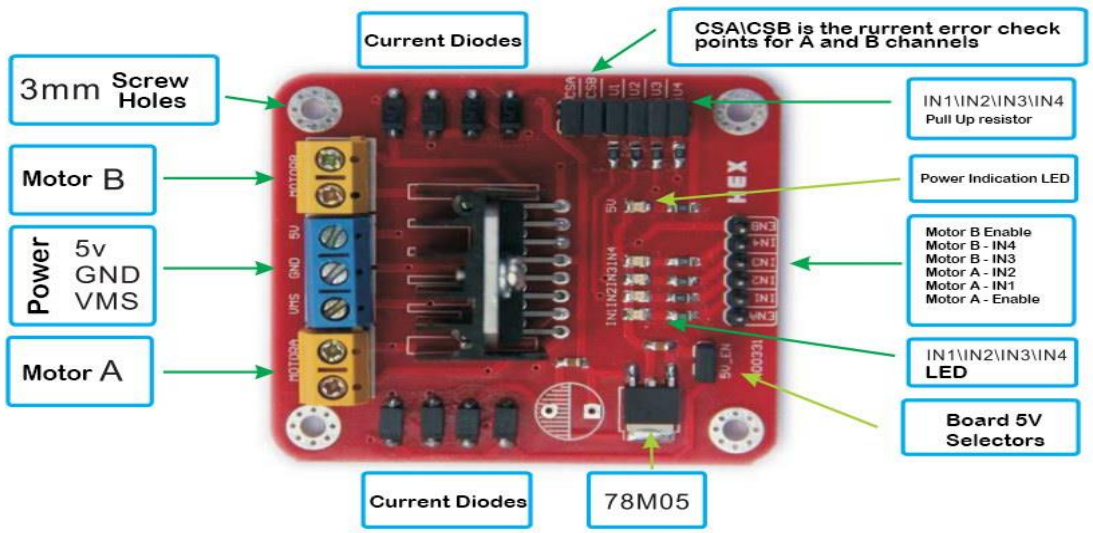
LIMITATIONS

- The sensors' may show poor performance near reflective objects.
- The device is not suitable for rough surfaces.
- DO NOT use any cleaning supplies on the device
- The device may contain small parts not suitable for children under age 5.

Supplier contact information:
E-mail: jcnelectronicscorporation@gmail.com

2. L298 DUAL H-BRIDGE MOTOR DRIVER MODULE SPECIFICATIONS

- Driver : L298
- Driver power supply : +5V~+46V
- Driver peak current : 2A
- Logic power output Vss : +5~+7V (internal supply +5V)
- Logic current : 0~36mA
- Controlling level: Low -0.3V~1.5V, high : 2.3V~Vss
- Enable signal level: Low -0.3V~1.5V, high : 2.3V~Vss
- Max drive power : 25W (Temperature 75 ℃)
- Working temperature : -25℃~+130℃
- Dimension : 60X54mm
- Driver weight : ~48g



3. CODES FOR ARDUINO 1

```
#include <SoftwareSerial.h>

//Sensor pins
  byte trigPinLow =  7;   //Lower sendor trig pin
  byte echoPinLow = 8;
  byte trigPinUp = 3;     //Upper sensor trig pin
  byte echoPinUp = 4;

//Motor pins
  //Directions (Straight movement R0&L0=0 R1&L1=1)
    int L0 = A0;
    int L1 = A1;
    int R0 = A2;
    int R1 = A3;
  //PWM
    byte pwmL = 5;
    byte pwmR = 6;

//Previous turn (0 no prev turn 1 right 2 left)
byte turnside = 0;

//CNY70 pins
  byte FL = 9;
  byte BL = 10;
  byte BR = 11;
  byte FR = 12;

//Communication pin
  byte com = 13;
```

```
//Reset pins
  int resetm = A7;
  int resetf = A6;

void setup() {

//Serial port
Serial.begin (9600);
Serial.println("Start");

//Interrupt pin
//  attachInterrupt(digitalPinToInterrupt(2), CNYCHECK, LOW);

//Sensor pins
  pinMode(trigPinLow, OUTPUT);
  pinMode(echoPinLow, INPUT);
  pinMode(trigPinUp, OUTPUT);
  pinMode(echoPinUp, INPUT);

//Motor pins
  //Directions
    pinMode (L0, OUTPUT);
    pinMode (L1, OUTPUT);
    pinMode (R0, OUTPUT);
    pinMode (R1, OUTPUT);
  //PWM
    pinMode (pwmL, OUTPUT);
    pinMode (pwmR, OUTPUT);
  //Initializations
    analogWrite (pwmR, 0);
    analogWrite (pwmL, 0);
```

```
//CNY70 pins
  pinMode (FL, INPUT);
  pinMode (BL, INPUT);
  pinMode (BR, INPUT);
  pinMode (BL, INPUT);

//Communication pin
  pinMode (com, OUTPUT);
  digitalWrite (com, LOW);

//Reset pins
  pinMode (resetf, OUTPUT);
  analogWrite(resetf,0);

}

void(* resetFunc) (void) = 0;//declare reset function at address 0

void loop() {

Serial.println("Restart");

FINDGARBAGE();
WAIT();       // open catch
GOGARBAGE();
WAIT();       // close catch
FINDFENCE();
WAIT();       // drop
NEXTPREPARE();
}
```

```
void CNYCHECK () {
  STOP();
  int cnyval = digitalRead(FL)+2*digitalRead(BL)+4*digitalRead(BR)+8*digitalRead(FR);
  randomSeed(millis());
  int wai = random(100,200);
  switch(cnyval){

    //Front cases
    case 1 :
      TURNRIGHT();
      delay(wai);
      STOP();
      break;
    case 8 :
      TURNLEFT();
      delay(wai);
      STOP();
      break;
    case 9 :
      GOBACK();
      delay(300);
      if(random(1)) TURNRIGHT();
      else TURNLEFT();
      delay(wai);
      STOP();
      break;

    //Back cases
    case 2 :
      GOSTRAIGHT();
```

```
    delay(100);
    TURNRIGHT();
    delay(wai);
    GOSTRAIGHT();
    delay(300);
    STOP();
    break;
   case 4:
    GOSTRAIGHT();
    delay(100);
    TURNLEFT();
    delay(wai);
    GOSTRAIGHT();
    delay(300);
    STOP();
    break;
   case 6:
    GOSTRAIGHT();
    delay(100);
    if(random(1)) TURNRIGHT();
    else TURNLEFT();
    delay(wai);
    GOSTRAIGHT();
    delay(300);
    STOP();
    break;
  }
}

boolean DISTANCECOMPARE(){
  Serial.println("Distance compare");
```

```
 //each cm is 30
 //return 0 nothing
 //return 1 garbage detect

byte garbage=0;    //Garbage detection
if(DISTANCELOW()<750){
  for(int i=0 ; i<10 ; i++){
   long low = DISTANCELOW();
   delay(100);
   long up = DISTANCEUP();
   delay(100);
   if (abs(low-up) > 120) garbage++;
   }

  if (garbage > 6) return HIGH;
  else return LOW;
 }
}

long DISTANCELOW (){
 long data=0;

 digitalWrite(trigPinLow, LOW);
 delayMicroseconds(2);

 digitalWrite(trigPinLow, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPinLow, HIGH);
 data=pulseIn(echoPinLow, HIGH)/2;
 return data;
```

```
}

int DISTANCELOWAVG () {
  Serial.println("Distancelowavg");
  int a=0;
  for (int i=0; i<3; i++){
    a=a+DISTANCELOW();
  }
  a=a/3;
  Serial.println(a);
  return a;
}

long DISTANCEUP (){
  long data=0;

  digitalWrite(trigPinUp, LOW);
  delayMicroseconds(2);

  digitalWrite(trigPinUp, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPinUp, LOW);
  data=pulseIn(echoPinUp, HIGH)/2;
  return data;
}

void FENCEALIGN(){
  noInterrupts();
  GOSTRAIGHT();
  while (digitalRead(FL)||digitalRead(FR)) STOP();
  do{
```

```
    if (digitalRead(FL)){
      TURNLEFT();
      while(!digitalRead(FR)){}
      STOP();
    }
    if (digitalRead(FR)){
      TURNRIGHT();
      while(!digitalRead(FL)){}
      STOP();
    }
  }while (!(digitalRead(FL)&&digitalRead(FR)));
}

void FENCEDETECT(){
  Serial.println("Fence detect");
  delay(500);

  randomSeed(millis());
  int a = random (200);
  long up;

  if (a>100){
    if (DISTANCEUP()>1200){
      TURNRIGHT();
      while(DISTANCEUP()>1200) {}
      STOP();
    }
    up = DISTANCEUP();
    do{
      TURNRIGHT();
      delay(50);
```

```
    STOP();
   }while(DISTANCEUP()>=up);
   do{
    up = DISTANCEUP();
    TURNRIGHT();
    delay(50);
    STOP();
   }while(DISTANCEUP()<up);
  }

  else{
   if (DISTANCEUP()>1200){
    TURNLEFT();
    while(DISTANCEUP()>1200) {}
    STOP();
   }
   up = DISTANCEUP();
   do{
    TURNLEFT();
    delay(50);
    STOP();
   }while(DISTANCEUP()>=up);
   do{
    up = DISTANCEUP();
    TURNLEFT();
    delay(50);
    STOP();
   }while(DISTANCEUP()<up);
  }
 }
```

```
void FINDFENCE(){
  Serial.println("Find fence");
  FENCEDETECT();
  //FENCEALIGN();
}

void FINDGARBAGE(){
  Serial.println("Find garbage");
  turnside=0;
  while(!GARBAGEDETECT()){
    RANDOMTRAVEL();
  }
  GARBAGEALIGN();
  if(DISTANCELOWAVG()>400){
    analogWrite (resetf,255);
    delay(10);
    analogWrite(resetf,0);
    resetFunc();
  }
}

void GARBAGEALIGN(){
  Serial.println("Garbage allign");
  delay(500);
//  noInterrupts();
  if(DISTANCELOWAVG()<300){
    while (DISTANCELOWAVG()<300){
      GOBACK();
      delay(50);
      STOP();
      //if((digitalRead(BL)||digitalRead(BR))) digitalWrite(reset,HIGH);
```

```
    }
   STOP();
  }
//  interrupts();
}

boolean GARBAGEDETECT(){
  Serial.println("Garbage detect");

  if(DISTANCECOMPARE()){

  if(turnside==1){
    if (DISTANCELOWAVG()<200){
      TURNRIGHT();
      delay(60);
    }
    else{
      TURNRIGHT();
      delay(30);
    }
    STOP();
  }
  if(turnside==2){
    if (DISTANCELOWAVG()<200){
      TURNLEFT();
      delay(60);
    }
    else{
      TURNLEFT();
         delay(30);
    }
```

```
   STOP();
  }

   if(DISTANCELOWAVG()<100){
 //    noInterrupts();
     GOBACK();
     delay(20);
     STOP();
//     interrupts();
   }
   else {
//     noInterrupts();
     GOSTRAIGHT();
     delay(20);
     STOP();
//     interrupts();
   }
  if(DISTANCECOMPARE()) return HIGH;
  else return LOW;
 }
 else return LOW;
}

void GOBACK (){
 Serial.println("Go back");
 delay(500);
     digitalWrite (R0, HIGH);
     digitalWrite (R1, LOW);
     digitalWrite (L0, HIGH);
     digitalWrite (L1, LOW);
```

```
    analogWrite (pwmR, 245);
    analogWrite (pwmL, 255);

}

void GOGARBAGE(){
 Serial.println("Go garbage");
 delay(500);
 int a = DISTANCELOWAVG();
 if(a>200 && a<750){
  while(a>200 && a<750){
    GOSTRAIGHT();
    delay(20);
    STOP();
    a = DISTANCELOWAVG();
  }
 }
 if(a>50 && a<=200){
  while(a>50 && a<300){
    GOSLOW();
    delay(20);
    STOP();
    a = DISTANCELOWAVG();
  }
  Serial.println("closest ");
  Serial.print(a);
 }
 if (a>50){
  analogWrite (resetf,255);
  delay(10);
  analogWrite (resetf,0);
```

```
    Serial.println("resetf");
    resetFunc(); //call reset
  }
}

void GOSLOW(){
    Serial.println("Go slow");
      digitalWrite (R0, LOW);
      digitalWrite (R1, HIGH);
      digitalWrite (L0, LOW);
      digitalWrite (L1, HIGH);

      analogWrite (pwmR, 100);
      analogWrite (pwmL, 100);
}

void GOSTRAIGHT () {
  Serial.println("Go straight");
  delay(500);
      digitalWrite (R0, LOW);
      digitalWrite (R1, HIGH);
      digitalWrite (L0, LOW);
      digitalWrite (L1, HIGH);

      analogWrite (pwmR, 245);
      analogWrite (pwmL, 255);
}

void NEXTPREPARE(){
  Serial.println("Next prepare");
  delay(500);
```

```
  GOBACK();
  delay(200);
  STOP();
  randomSeed(millis());
  int a = random(0,200);
   if(a>100){
    TURNRIGHT();
    delay(100);
    STOP();
   }
   else{
    TURNLEFT();
    delay(100);
    STOP();
   }
//  interrupts();
  RANDOMTRAVEL();
}

void RANDOMTRAVEL(){
  Serial.println("Random Travel");
  /*
   int turnside
   0 gostraight chosen
   1 turnright chosen
   2 turnleft chosen
  */
   randomSeed(millis());
   int a = random(0,250);
   switch(turnside){
    case 0 :
```

```
 if (a<30){
  GOSTRAIGHT();
  delay(100);
  STOP();
  turnside=0;
 }
 else{
  if(a>170){
   TURNRIGHT();
   delay(30);
   STOP();
   turnside = 1;
  }
  else{
   TURNLEFT();
   delay(30);
   STOP();
   turnside = 2;
  }
 }
break;
case 1 :
 if (a<30){
  GOSTRAIGHT();
  delay(100);
  STOP();
  turnside=0;
 }
 else{
  TURNRIGHT();
  delay(30);
```

```
      STOP();
      turnside = 1;
     }
   break;
   case 2 :
    if (a<30){
      GOSTRAIGHT();
      delay(100);
      STOP();
      turnside=0;
     }
     else{
      TURNLEFT();
      delay(30);
      STOP();
      turnside = 2;
     }
   break;
  }
}

void STOP (){
    digitalWrite (R0, LOW);
    digitalWrite (R1, LOW);
    digitalWrite (L0, LOW);
    digitalWrite (L1, LOW);

    analogWrite (pwmR, 0);
    analogWrite (pwmL, 0);
}
```

```
void TURNLEFT () {
  Serial.println("Turn left");
  delay(500);
     digitalWrite (R0, LOW);
     digitalWrite (R1, HIGH);
     digitalWrite (L0,HIGH);
     digitalWrite (L1, LOW);

     analogWrite (pwmR, 245);
     analogWrite (pwmL, 255);
}

void TURNRIGHT () {
  Serial.println("Turn right");
  delay(500);
     digitalWrite (L0, LOW);
     digitalWrite (L1, HIGH);
     digitalWrite (R0,HIGH);
     digitalWrite (R1, LOW);

     analogWrite (pwmR, 245);
     analogWrite (pwmL, 255);
}

void WAIT(){
  Serial.println("Wait");
  delay(500);
  digitalWrite(com,HIGH);
  digitalWrite(com,LOW);
  delay(100);
  pinMode(com,INPUT);
```

```
  do{
  }while(digitalRead(com)==LOW);
  pinMode(com, OUTPUT);
  digitalWrite(com,LOW);
}
```

4. CODES FOR ARDUINO 2

```
#include <Servo.h>

//Catch servos
  Servo cf;
  Servo cb;

//Lift servos
  Servo lb;
  Servo lf;

//Platform servo
  Servo p;

//Communication pin
  byte com = 13;


void setup() {
  delay(1000);
```

```
//Interrupt pin
  attachInterrupt(digitalPinToInterrupt(2), RESET, HIGH);
//Catch servos
  cf.attach(3);
  cb.attach(4);
    //initializations
    cf.write(10);
    cb.write(25);

//Lift Servos
  lb.attach(10);
  lf.attach(11);
    //Initializations
      lb.write(45);
      lf.write(130);

//Platform servo
  p.attach(12);
    //Initialization
      p.write(20);

//Communication pin
  delay (100);
  pinMode (com, INPUT);
}

void(* resetFunc) (void) = 0;//declare reset function at address 0

void loop() {
```

```
WAIT1();
  OPENCATCH();
WAIT2();
WAIT1();
  DOWNPLATFORM();
  CLOSECATCH();
  UPPLATFORM();
WAIT2();
WAIT1();
  UPLIFT();
  DROPPLATFORM();
  DOWNLIFT();
WAIT2();
}

void RESET (){
  resetFunc(); //call reset
}

void CLOSECATCH (){
    for (int i=157;i>5;i--){
      cf.write(5+i);
      cb.write(20+47*i/50);
      delay(50);
    }
}

void DOWNLIFT (){
          for(int i=80;i>15;i--){
```

```
            lb.write(30+i);
            lf.write(148-8*i/7);
            delay(50);
          }
}

void DOWNPLATFORM (){
  for(int i=20;i<80;i++){
    p.write(i);
    delay (50);
  }
}

void DROPPLATFORM(){
    p.write(100);
    delay(500);
    p.write(20);
}

void OPENCATCH (){
    for (int i=5;i<157;i++){
      cf.write(5+i);
      cb.write(20+47*i/50);
      delay(50);
  }
}

void UPLIFT () {
          for(int i=15;i<80;i++){
```

```
            lb.write(30+i);
            lf.write(148-8*i/7 );
            delay(50);
          }
}

void UPPLATFORM (){
  for(int i=80;i>20;i--){
    p.write(i);
    delay (50);
  }
}

void WAIT1 (){
  do{} while(digitalRead(com)==LOW);
  pinMode(com,OUTPUT);
  digitalWrite(com,LOW);
}

void WAIT2 () {
  digitalWrite(com,HIGH);
  digitalWrite(com,HIGH);
  delay(100);
  pinMode(com,INPUT);
}
```